

REMARKS/ARGUMENTS

Entry and consideration of these Remarks and reconsideration of all claims remaining of record as presently amended are respectfully requested.

The Examiner is also requested to formally acknowledge the receipt and entry of the Applicant's amendments to drawing Figures 2 and 11 that were previously submitted in Applicant's September 20, 2007 Response.

The rejection of claims 1, 2, 4-7 and 9-11 under 35 U.S.C. §103(a) as allegedly being unpatentable over Ohno et al. (U.S. Patent No. 5,609,525) in view of a publication by Smith entitled, "The Scientist and Engineers Guide to Digital Signal Process, Circular Buffering" and further in view of Carrel et al. (U.S. Patent No. 5,778,167) is respectfully traversed.

Ohno et al. describe an arrangement in a video game machine that uses a "save" memory that is write protected unless a game data save operation is requested for a selected game having the same title as the game being stored – so that saved games of other titles are not inadvertently overwritten, for example, when saving data of an unfinished game that is terminated mid-play. (See e.g., the '525 patent at col. 2, lines 52-63 and Fig. 10, blocks ST2-ST10.) If the title of the game for the game data being saved is found to already exist in the "save" memory then that memory area is overwritten with the current game data being saved. (See '525 patent at col. 5, lines 66 through col. 6, line 3.) Applicant respectfully submits that Ohno et al.'s disclosed memory write protection arrangement is in no manner similar in operation or even remotely suggestive of Applicant's claimed method and implementation. Ohno et al. teaches a write protection arrangement intended for use with an SRAM and does not disclose the use of electrically erasable/rewritable non-volatile memory or discuss the problems inherent in using such. As pointed out in Applicant's specification, electrically rewritable non-volatile memory

has the serious problem that electrically erasable/rewritable memory elements in the memory typically have a shorter and somewhat unpredictable useful lifespan when subjected to frequent and repetitive use as compared with memory elements of other types of memory such as static RAM or SRAM. If a particular backup portion/area in an electrically erasable/rewritable memory is continually being rewritten, as happens when using/playing video and electronic games, the life of the storage elements in that backup area/portion of memory become shorter and less reliable than another other less used areas/portions of the memory. As such, as memory areas in an electrically erasable/rewritable memory become unreliable or go bad, the unreliable/bad areas can be flagged to prevent further use and only other areas that are still reliable within that memory will be used. Consequently, when using an electrically erasable/rewritable non-volatile memory in gaming devices, the possibility of permanently and unpredictably losing data that was previously saved/stored always exists whenever an attempt is made to write into the memory. Since Ohno does not use electrically erasable/rewritable memory, Ohno does not address such issues and provides no solution to the problem where if an attempt is made to write to an electrically erasable/rewritable memory at a memory location/area that contains the last remaining set of previously saved game data, and that attempt fails because of a failure of a memory element at that instant, then that previously saved data will be forever lost. In contrast to Ohno, Applicant's claims set forth a method and implementation for specifically overcoming this potential loss of a last remaining set of previously saved game data *in a game device that uses an electrically erasable non-volatile memory device.*

More specifically, as currently amended, Applicant's independent claims 1, 5, 10 and 11 provide a game data backup arrangement wherein a failure of a memory element occurring in an electrically erasable non-volatile memory does not result in a loss of older game data stored in a

last remaining available backup area and, thus, ensures that the backup memory will always contain at least one storage area containing older game data even when one or more memory areas have failed because further writing into the memory is prohibited under conditions wherein an attempt to write the latest game data to the selected write-objective backup area is unsuccessful after a predetermined number of attempts have been made *and* only a backup area containing older game data saved immediately before the last latest game data is available as a write-objective backup area.

Applicant also asserts that the claimed implementation does not operate or function in the manner of a “circular buffer” memory, nor is the operation of a conventional circular buffer memory in any way remotely suggestive of applicant’s claimed implementation. In particular, Applicant’s claimed implementation evaluates or compares the write age of each memory backup storage area to determine the area with the oldest write age, whereas a circular buffer uses a writer pointer *that merely steps incrementally and sequentially through areas where data may be stored without checking the age of the data stored therein*. For example, in Smith’s circular buffer, if a memory element/area goes bad or becomes unwritable, the next sequential memory area will be written into with new data overwriting the oldest data – even if that next sequential memory area is the last remaining working memory area which contains older data. In such a situation, unlike Applicant’s claimed arrangement, the older data will be permanently lost because it will be overwritten as there is no mechanism in a circular buffer to recognize this situation and prohibit the overwriting.

Moreover, Applicant’s claims require that writing is prohibited only when two specific conditions together exist at the same time, namely: 1) a predetermined number of unsuccessful write attempts must have been performed; and 2) only a backup area containing older game data

saved from a previous gaming session is available as a write-objective backup area. Neither Ohno et al. nor Smith describing the operation of a conventional “circular buffer”, nor Carrel et al., nor any of the prior art of record art of record, either teaches or suggests this specific arrangement in an apparatus or a method for backing up game data as presently set forth in Applicant’s independent claims. Accordingly, it is submitted that independent claims 1, 5, 10 and 11, as presently amended, are patentably distinct over the teachings of Ohno et al. in view of Smith and further in view of Carrel et al.

Claims 2, 4, 6, 7 and 9 are dependent on either claims 1 or 5 and since neither Ohno et al. nor the operation of a conventional “circular buffer” nor any of the prior art of record suggest the features or steps as discussed above and set forth in Applicant’s independent claims 1 or 5 as amended, it is respectfully submitted that these dependent claims are patentable over the individual as well as the combined teachings of these references.

All of the above and foregoing amendments to specification and claims are fully supported by the original disclosure and, thus, no new matter has been added. If the Examiner should disagree, however, it is respectfully requested that the challenged limitation be pointed out with particularity in the next Action so that support may be cited in response.

In view of Applicant’s foregoing amendments and remarks, it is believed that the application is in condition for allowance. Favorable consideration and allowance of this application are respectfully solicited. If any small matter remains outstanding, the Examiner is encouraged to telephone Applicant’s representative at the telephone number listed below.

Response/Amendment After Final Rejection
U.S. Application No. 10/712,312

Atty. Docket No.: 723-1451
Art Unit No.: 3711

The Commissioner is hereby authorized to charge the undersigned's deposit account #14-1140 in whatever amount is necessary for entry of these papers and the continued pendency of the captioned application.

Respectfully submitted,

NIXON & VANDERHYE P.C.

By:



William G. Niessen

Reg. No. 29,683

WGN/edg
901 North Glebe Road, 11th Floor
Arlington, VA 22203-1808
Telephone: (703) 816-4000
Facsimile: (703) 816-4100

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of Confirmation No. 8866
Hitoshi YAMAGAMI Atty. Ref.: 723-1451
Appl. No. 10/712,312 TC/A.U. 3714
Filed: November 14, 2003 Examiner: Meagan J. THOMASSON
For: METHOD AND APPARATUS FOR BACKING-UP GAME DATA

* * * * *

November 24, 2008

MAIL STOP AF
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

SUBSTITUTE SPECIFICATION

(MARKED-UP VERSION)

TITLE OF THE INVENTION

METHOD AND APPARATUS FOR BACKING-UP GAME DATA

FIELD OF THE INVENTION

5 The present invention relates to a game machine, a game backup control program and a game backup control method for backing-up game data. More specifically, the present invention relates to a game machine provided with a nonvolatile memory having two or three or more backup areas within an electrically rewritable storing area and game data is written into the backup area, and a backup control program and a backup control method for the game data.

10

BACKGROUND

An example of this kind of a conventional game machine is disclosed in a Japanese Laid-open Patent No. 5-12139 published on January 22, 1993. In a hot start system of the prior art, a backup portion is selected among storing areas for game 1 to game 3, provided in a static 15 RAM of a game cartridge, in response to an operation by an operator. If no game data has been backed-up in a selected backup portion, a backup-objective data area of the scratch RAM provided in a video game machine is initialized so as to start a game at the beginning. On the other hand, if the game data has been backed-up in the selected backup portion, the game data is read into the scratch RAM of the video game machine so as to resume the game. As the 20 game progresses, the game data in the scratch RAM is renewed, and for example, every time of making a hero act, the backup-objective data out of the game data of the scratch RAM is automatically stored in a predetermined (selected) storing area in the static RAM of the game cartridge, whereby the game status can be automatically backed-up at any time.

Furthermore, another example of the prior art is disclosed in a Japanese Patent
25 Laid-open Patent Publication No. 10-31611, published on February 3, 1998. In a file system for a nonvolatile memory storage medium disclosed in this prior art, a logical block number obtained by searching a minimum writing times information value among unassigned state

information in a usage frequency list is made as a write-objective block allowing, available memory blocks to be written uniformly, that is, the frequencies of the writing to the blocks are made even so as to effectively lengthen the life of the nonvolatile memory.

However, in the former, a static RAM is utilized as a backup memory backing-up the game data, and therefore, it is necessary to utilize a battery for. Thus, there is a significant possibility that the battery will die during a game in which the battery is low, and in such a case, there is a problem that all the game data within the static RAM may be lost. Recently, in order to avoid such problems, nonvolatile memory (e.g., flash memory, ferroelectric memory, etc.), which does not need a battery for holding data, is being utilized as a backup memory. Thus, with utilizing the nonvolatile memory, there is no fear of battery exhaustion. However, flash memory, has a disadvantage of being slow in writing speed and short in life as compared with static RAM. Therefore, if a backup portion selected by the user is continually being rewritten, the life of the storing elements in that backup area/portion of memory becomes shorter than another backup area/portion, and therefore, another problem occurs in that the game data stored in that continually rewritten backup portion of memory is apt to be lost.

On the other hand, in the latter example, the number of rewriting times of each memory block of the nonvolatile memory is made the same so as to prevent a specific memory block from being extremely shorter in life than other memory blocks.

However, even in view of the above-described prior arts, there still exists problems that if a storing element (or a part of storing area) becomes abruptly defective (due to a short life) during the time the last-latest game data is being written in the backup portion, it becomes impossible to write the last-latest game data, and furthermore, in the instance that a part of the last-latest game data has been written onto older game data, the older game data may also be lost. Furthermore, in a case that the power to the game machine is turned-off midway during writing the last-latest game data, there occurs a problem that both the last-latest game data and the older game data are lost, or worse yet, all the game data is lost. It is noted that preventing/loss of game data obtained by progressing through a game for long hours is critical

for the player.

SUMMARY OF THE INVENTION

Notable aspects of the illustrative exemplary implementations disclosed herein include:

- 5 a novel game machine apparatus, a game data backup control program and backup control method.

Another aspect of the example implementation disclosed herein is to provide a game machine, a game data backup control program and a backup control method capable of reliably preventing game data from at least the previous game play session from being lost.

10 An illustrative exemplary non-limiting implementation of a game machine comprising a nonvolatile memory which has two or three additional backup areas in an electrically rewritable storing area and which writes game data in these backup areas. The game machine comprises an area selecting means for preferentially selecting, whenever last-latest game data is about to be written, a backup area having an older writing time among two or three more 15 backup areas as a write-objective backup area for the last-latest game data; a writing means for executing a writing of the last-latest game data into the write-objective backup area selected by the area selecting means; a determining means for determining whether or not the writing of the last-latest game data could be performed by the writing means; a repeating means for repeating, when it is determined the writing of the last-latest game data could not be performed by the 20 determining means, a selection of the write-objective backup area by the area selecting means as necessary; and a prohibiting means for prohibiting, when only a backup area stored with game data written immediately before the last-latest game data finally becomes a selectable state by the area selecting means, a writing to the write-objective backup area.

Specifically, the game machine (10: reference numeral corresponding to a preferable 25 embodiment described later and so forth) is provided with the nonvolatile memory (58) having at least the two backup areas in the electrically rewritable storing area, and the game data is written into the backup area. The area selecting means (40, S15) preferentially selects, when

the last-latest game data is to be written, the backup area stored with the game data having an older writing time among at least two backup areas as the write-objective backup area of the last-latest game data. The writing means (40, S17, S25) executes the writing of the last-latest game data into the write-objective backup area selected by the area selecting means (40, S15).

5 The determining means (40, S27) determines whether or not the writing of the last-latest game data could be performed by the writing means (40, S17, S25). The repeating means (40, S101, S103) repeats, whenever it is determined that the writing of the last-latest game data could not be performed by the determining means (40, S27), the selection of the write-objective backup area by the area selecting means (40, S15) as necessary. More specifically, in the case where
10 the number of the backup areas is only two, even if it is determined that the last-latest game data could not be written, the selection of the write-objective backup area by the area selecting means (40, S15) is not repeated. On the other hand, in the case where the number of the backup areas is three or more, when it is determined the last-latest game data could not be written, the selection of the write-objective backup area by the area selecting means (40, S15) is repeated
15 so as to repeatedly execute the writing process of the last-latest game data. Then, the prohibiting means (40, S29) prohibits, when only the backup areas stored with the game data written immediately before the last-latest game data finally becomes the selectable state by the area selecting means (40, S15), the writing to the write-objective backup area.

It is noted that in a situation where the game data has not previously been stored in the
20 backup area, the backup area is selected with priority as a write-objective backup area.

One additional feature of the non-limiting example implementation disclosed herein is that it makes it possible to securely leave stored in memory the game data saved from a previous generation of game play.

In one non-limiting example implementation of the game machine disclosed herein, the
25 writing means writes historical data for discriminating between oldness and newness of the game data by being included in the last-latest game data, and the area selecting means selects, before writing the last-latest game data, a backup area stored with game data written earlier

than the last-latest game data as the write-objective backup area on the basis of the historical data. More specifically, since the writing means (40, S17, S25) writes the last-latest game data including the historical data for discriminating between the oldness and the newness of the game data, the area selecting means (40, S15) can select, before writing the last-latest game data, the backup area stored with the game data having an older writing time as the write-objective backup area on the basis of the historical data included in the old data.

Accordingly, for example, game data having the oldest historical data is determined, allowing, the backup area in which the oldest game data is stored to be selected as the write-objective area.

In another non-limiting example implementation of the game machine disclosed herein, the prohibiting means includes a means for forcedly terminating a writing process of the last-latest game data when only the backup area stored with the game data written immediately before the last-latest game data finally becomes selectable as the write objective backup area by the area selecting means. More specifically, a terminating means (40, S29) forcedly terminates the writing process of the last-latest game data whenever only the backup area stored with the game data written immediately before the last-latest game data finally becomes selectable as a write-objective backup area by the area selecting means. Consequently, by this means, it is possible to securely leave the old game data saved in an area in memory for which overwriting is prevented. At this time, the last-latest game data is canceled.

In yet another non-limiting example implementation of a game machine disclosed herein, comprises a predetermined alarm message is displayed whenever overwriting of game data is prohibited. More specifically, when overwriting is prohibited by the prohibiting means (40, S29), a message displaying means (14, 40, S31) displays a predetermined alarm message so as to urge the user to exchange the nonvolatile memory.

One aspect of the exemplary implementation disclosed herein comprises a backup control arrangements in which the last-latest game data is written into two or more backup areas in an electrically rewritable storage area of a nonvolatile memory connected to the game

machine. In this example implementation, the backup control arrangement makes a computer (processor) of the game machine perform: an area selection process that, whenever last-latest game data is written, selects a backup area containing stored game data of an older write time among two or more backup areas as a write-objective for writing (saving) the last-latest game data; writing of the last-latest game data to the selected write-objective backup area; determining whether or not the writing of the last-latest game data could be performed; repeating the selection of the write-objective backup area as necessary, if it is determined that the writing of the last-latest game data could not be performed; and, prohibiting a writing to the write-objective backup area, when a backup area stored with game data written immediately before the last-latest game data becomes selectable.

Another aspect of the exemplary implementation disclosed herein includes a backup control method by which game data is written in two or three more backup areas in an electrically rewritable storing area of a nonvolatile memory connected to a game machine, and which comprises steps of: (a) selecting, as a write-objective backup area a backup area, which stores game data having an older writing time and to which the last-latest game data can be written; (b) canceling the writing of last-latest game data, when a writing into the write-objective backup area is not executable and only a backup area that was stored with game data immediately before the last-latest game data is selectable as a write-objective backup area, so as to leave intact the game data that was written immediately before the last-latest game data.

In the presently disclosed non-limiting example implementations of both the game data backup control apparatus and the backup control method, it becomes possible to securely leave game data from a previous generation of game play in memory, as is similarly the case for the above-described example game machine implementation.

The above non-limiting features, aspects and advantages of the example implementations described herein will become more apparent from the following detailed description when taken in conjunction with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is an illustrative view showing one example implementation of an appearance of a game machine;

5 Figure 2 is a block view showing an electrical configuration of the game machine shown in Figure 1;

Figure 3 is an illustrative view showing a memory map of a ROM provided in a game cartridge shown in Figure 2;

10 Figure 4 is an illustrative view showing a selection of game data backup areas in an example implementation wherein a flash memory as shown in Figure 2 is provided with two backup areas;

Figure 5 is an illustrative view showing a selection of backup areas for an example wherein a flash memory as shown in Figure 2 is provided with two backup areas;

15 Figure 6 is an illustrative view showing an example state in which two backup areas of a flash memory as illustrated by Figure 2 are provided, wherein one backup area is unwritable and the other backup area is prohibited from being written;

Figure 7 is an illustrative view showing one example of a save screen, a save end screen and an error screen as displayed on an LCD display device as shown in Figure 1 and Figure 2;

20 Figure 8 is an illustrative view showing a state in which two backup areas of a flash memory as illustrated by Figure 2 are provided and a writing prohibiting flag area is further provided, wherein one backup area is unwritable and the other backup area is prohibited from being written;

Figure 9 is a flowchart showing one example implementation of an overall process performed by a CPU shown in Figure 2;

25 Figure 10 is a flowchart showing an example backup process performed by the CPU shown in Figure 2;

Figure 11 is a flowchart showing an example backup area selection process of the backup areas of the CPU shown in Figure 2;

Figure 12 is a flowchart showing an example ~~renewal~~ historical data renewal process of historical data of the CPU shown in Figure 2;

Figure 13 is an illustrative view showing another non-limiting illustrative example of a memory map of a flash memory provided in a game cartridge;

5 Figure 14 is a flowchart showing a part of a backup process performed by the CPU for a memory map as shown in Figure 13;

Figure 15 is a flowchart showing another part of the backup process performed by the CPU for a memory map as shown in Figure 13; and

10 Figure 16 is an illustrative view showing another example of the memory map of the flash memory provided in the game cartridge loaded into the example game machine illustrated in Figure 13 embodiment.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENTS

Referring to Figure 1, a game machine 10 of one example implementation includes a
15 case 12. The case 12 is, on the surface thereof, provided with a color liquid crystal display 14 (hereinafter, referred to as "LCD") at an approximately center. The LCD 14 is displayed with a game space and game characters appearing in the game space, and messages as necessary. The case 12 is on its surface provided with operating buttons 16, 18, 20, 22, 24, 26 and 28. The operating buttons 16, 18 and 20 are placed at a left of the LCD 14, and the operating buttons 22
20 and 24 are placed at a right of the LCD 14. Furthermore, the operating buttons 26 and 28 are placed at an upper end surface (above the LCD) of the case 12.

The operating button 16 is a cross key which functions as a digital joystick and instructs a moving direction of a game character displayed on the LCD 14 and moves a cursor by operating any one of four depression portions. The operating button 18 is a start button formed by a push button and utilized for instructing a start of the game and etc. The operating button 20 is a select button formed by the push button and utilized for selecting a game mode and etc.
25

The operating button 22 is an A button formed by the push button and allows the game

character displayed on the LCD 14 to perform an arbitrary action such as hitting, throwing, catching, riding, jumping and etc. The operating button 24 is a B button formed by the push button, and utilized for changing to a game mode selected by the select button 20, canceling the action determined by the A button 22, and etc. The operating button 26 is a left depression button (L button) formed by the push button, and the operating button 28 is a right depression button (R button) formed by the push button. The operating buttons 26 and 28 can perform the same operation as the A button 22 and the B button 24 and also function as a subsidiary of the A button 22 and the B button 24.

The case 12 is, at an upper end of its rear surface, formed with a loading slot 30 into which a game cartridge 32 is loaded. Although not illustrated, connectors are respectively provided at a depth portion of the loading slot 30 and at an end portion of the game cartridge 32 in the loading direction, and when the game cartridge 32 is loaded in the loading slot 30, the two connectors are connected with each other. Therefore, the game cartridge 32 is allowed to be accessed by a CPU 40 of the game machine 10 (see Figure 2).

Furthermore, the case 12 is below the A button 22 and the B button 24 on its surface provided with a speaker 34 for outputting a BGM, voices or onomatopoeic sound of the character and so on during the game.

It is noted that although not illustrated, the case 12 is provided with an external expansion connector on its upper surface, a battery accommodating box on its rear surface and a power switch, a sound level volume, an earphone jack and etc. on its bottom surface.

An electrical configuration of the game machine 10 is shown in Figure 2. Referring to Figure 2, the game machine 10 is provided with the CPU 40 as described above. The CPU 40 is called as a computer, a processor or the like and performs an overall control of the game machine 10. The CPU or computer 40 is connected to a work memory 44, an external memory interface (I/F) 46, a controller I/F 48, a VRAM 50 and an LCD driver 52 via an internal bus (hereinafter, simply referred to as "bus").

The work memory 44 is utilized as a work area or a buffer area of the CPU 40. The

external memory I/F 46 is connected with the game cartridge 32 loaded in the loading slot 30 via the connector (not shown) as described above.

The controller I/F 48 is connected with a controller 54 including the cross key 16, the start button 18, the select button 20, the A button 24, the B button 26, the L button 30 and the R button 32, and therefore, an operation signal in response to an operation of these buttons is input to the CPU 40 via the controller I/F 48.

The VRAM 50 is rendered with game image data and character image data in accordance with instructions of the CPU 40. Furthermore, the LCD driver 52 reads the game image data and the character image data rendered on the VRAM 50 in accordance with instructions of the CPU 40 and displays a game screen and characters on the LCD 14.

Furthermore, the game cartridge 32 is provided with a ROM 56 and a flash memory 58 of one example of a nonvolatile memory, and although not illustrated, the ROM 56 and the flash memory 58 are connected with each other via a bus and connected to connectors.

Accordingly, as described above, when the game cartridge 32 is loaded into the game machine 15 10, the CPU 40 is electrically connected to the ROM 56 and the flash memory 58.

It is noted that although the flash memory 58 is utilized in this embodiment, as another nonvolatile memory, a ferroelectric memory (FeRAM), EEPROM and etc. can also be utilized.

The ROM 56 is previously retained (stored) with a game program 560, image data (game image data, character image data, message display image data and etc.) 562, a backup control program 564 as shown in Figure 3. The backup control program 564 is constructed by comprises an area selecting program 564a, a trouble determining program 564b, a overwrite prohibiting program 564c, a message displaying program 564d and a historical data renewing program 564e. It is noted that these programs 564a to 564e are not executed independently but executed as a series of processes (see Figure 10).

25 It is noted that although not illustrated, the ROM 56 is stored with various data such as sound data and etc. required for the game and programs required therefore.

In addition, the flash memory 58 is stored with game data (backup data). For example,

when a user plays the game by utilizing the game machine 10, game data is stored (renewed) in the work memory 44 of the game machine 10 in accordance with the progress of the game. Then, the game data is written (saved) into the flash memory 58 of the game cartridge 32 in response to an instruction of the user or at a predetermined timing (event). Thereafter, in a case 5 the game is to be continued, the game is advanced as it is, and then, storing and backup of the game data described above are executed. Furthermore, in a case of ending the game once, the user reads the game data backed-up the last time (*i.e., the last play session*) and starts or resumes next time where the user left off.

The flash memory 58, as compared with a memory such as SRAM, has an advantage of 10 requiring no backup power source but has disadvantage of being short in rewritable life and in rewriting speed. Especially, at a time of writing data, when unwritable problems such as physical breakdown, electrical breakdown and etc. occur, a part of the game data cannot be written and hence, ~~breakage loss~~ of the game data occurs. Especially, ~~the In particular, since~~ game data is frequently rewritten as ~~the a~~ game progresses as described above, and therefore, 15 without taking preventive measures, the ~~breakage chance for loss~~ of the game data clearly exists.

It is noted that the physical breakdown includes a memory cell (storage element) breakdown, a soldering defect, a wiring defect and etc. The electrical breakdown includes a shutdown due to carelessness, an instant power failure and etc. In a case of the instant power 20 failure, a problem of not returning to a restoring process and etc. occurs.

In one example implementation, the flash memory 58 is provided with two backup areas in which the game data is sequentially stored. Thus, even if one backup area is broken down, the other backup area ~~surely retains~~ may be used to retain the game data, and whereby, it is possible to prevent all the data from being completely lost. Furthermore, when ~~only thing to~~ 25 the only operation which may be done is to perform an overwrite to game data (old game data) written immediately before, overwriting on the old data written immediately before is prohibited prevented (prohibited). Thus, a loss of the game data due to a trouble problem

occurring during overwriting the last writing of the latest game data onto over the old game data can be prevented.

It is noted that the flash memory 58 generally needs to be rewritten with data sector (block) by sector (block), but some products are rewritable address by address, and in this embodiment, the backup area includes one or two more sectors. That is, the number of the sectors included in the backup area is determined depending upon a volume of the game data.

The flash memory 58 has a first backup area 58a and a second backup area 58b as shown in Figure 4(A). Furthermore, as described above, the game data includes (is added with) historical data so as to sequentially write the game data to the first backup area 58a and the second backup area 58b in this embodiment. That is, Figure 4(A) shows a state that game data having historical data (1) is written to the first backup area 58a, and game data having historical data (2) is written to the second backup area 58b.

It is noted that it is also possible to determine which is the older game data by storing a pattern of saved game data in the work memory 44 in place of the historical data, for example.

Herein, in a case situation where the game data (last game data) that is stored in the work memory 44 of the game machine 10 is written to the flash memory 58 as “latest game” data, the historical data (3) is assigned to the last this latest game data, and an area to which the last this latest game data is to be written is selected between the first backup area 58a and the second backup area 58b according to a predetermined rule (condition for area selection) (condition for area selection).

The historical data is assigned to the game data in the order of 1 → 2 → 3 → 4 → 1 by the CPU 40 of the game machine 10, and included in the game data. For example, although not illustrated, the historical data is determined by a count value of a 2-bit counter which is incremented at a time of storing the game data. That is, the count value is varied from one another, “00” (“1”) → “01” (“2”) → “10” (“3”) → “11” (“4”), and the counter is reset at a time of a maximum value “11” (“11” → “00”).

It is noted that the maximum value “11” (“4”) is determined to be multiples of the

number of the backup areas. Accordingly, it is appropriate that a counter capable of counting at least the multiples of the number of the backup areas is utilized as the counter.

Furthermore, the condition for area selection is represented by four inequalities of $2 > 1$, $3 > 2$, $4 > 3$, $1 > 4$. These numerals 1 to 4 are numerical values indicated by the historical data and mean that the game data including the historical data of a numeral on the right side is older than the game data including the historical data of a numeral on the left side (written to the flash memory 58 earlier). Such the condition for area selection is determined (defined) by inequalities (conditions) utilized in correspondence to the historical data of the last “latest game” (i.e., latest saved) game data, and specifically represented by ~~a table~~ + table 1.

10

Table 1

Historical Data of <u>Last-Latest Game Data</u>	Condition for Area Selection
1	$4 > 3$ (3 is older than 4)
2	$1 > 4$ (4 is older than 1)
3	$2 > 1$ (1 is older than 2)
4	$3 > 2$ (2 is older than 3)

Accordingly, in Figure 4(A) state, as a write-objective area for saving the last-latest game data including the historical data (3), the first backup area 58a stored with the game data having an older writing time is selected according to the condition of the area selection ($2 > 1$).
15 The same is true for the following examples so as to select the write-objective backup area. Then, when the last-latest game data is written to the first backup area 58a, a state becomes like Figure 4(B).

In a Figure 4(B) state, in ~~a case of writing~~ which shows the next time saving of game data (last-latest game data) is performed, the historical data (4) is assigned to the last-latest game data obtained from game data stored in the work memory 44, and as ~~the~~ a write-objective area for the storing this last-latest game data, the second backup area 58b is selected according to the condition of ~~the~~ for area selection ($3 > 2$). When ~~the~~ Once this last-latest game data

including the historical data (4) is written to the second backup area 58b, a state is like that shown in Figure 5(A).

In a Figure 5(A) state, in a case of writing next which shows the next saving of game data (last i.e. the next time the latest game data is saved), the historical data (1) is assigned to the last game data stored in the from work memory 44, and as a write-objective area for the storing this last latest game data, the first backup area 58a according to the condition of the for area selection ($4 > 3$) is selected. When the last latest game data including the historical data (1) is written to the first backup area 58a, a state becomes like Figure 5(B).

In Figure 5(B) state, in a case of writing next which shows the next saving of game data (last game data), the historical data (2) is assigned to the last latest game data stored in the from work memory 44, and as the a write-objective area for the last storing this latest game data, the second backup area 58b according to is selected in accordance with the condition of the area selection ($1 > 4$) is selected. Then, the last this latest game data including the historical data (2) is written to the second backup area 58b.

Thus, although the writing (overwriting) of the game data is executed according to the condition for area selection, in some cases the last latest game data cannot be written to the area selected according to the condition for area selection due to the physical breakdown and the electrical breakdown. In this case, although it is probable that overwriting is performed with respect to the game data immediately before (one generation ago), if the physical breakdown or the electrical breakdown occurs during the overwriting, all the game data may be lost and therefore, the overwriting is prohibited with respect to the last game data in this embodiment. That is, a writing process of the game data is forcedly terminated.

Specifically, as shown in Figure 6, in a case where the game data including the which includes historical data (1) is written to into the first backup area 58a and the game data including the which includes historical data (4) is written to into the second backup area 58b, as the write-objective area for the last latest game data including the (stored in memory 44) which includes historical data (2), the second backup area 58b is selected according to the condition

of the area selection (1>4). Then, a writing process of the last for writing this latest game data is executed. In the ~~this~~ writing process of the game data, it is determined whether or not the game data is ~~written~~ normally written by ~~using~~ a checksum, and in a case the instance that a normal writing is not performed, the writing process is repeated by ~~a~~ plural number of times (e.g., three times). As a consequence of ~~repeating~~ the writing process plural times, if failing in the ~~a~~ failure of a normal writing of the game data ~~still occurs~~, it is determined to be then that backup area is designated as unwritable. Then, the writing process of the game data is forcedly terminated so as to prohibit ~~prevent~~ the writing of the last game data to another ~~the other~~ remaining backup area (first backup area 58a in Figure 6).

Furthermore, in a case the game data cannot be written to the flash memory 58, an alarm message is displayed on the LCD 14 so as to urge the user to exchange the flash memory 58 (to repair the game cartridge 32). For example, when the user instructs to store the game data (display a save screen), in response thereto a screen for selecting whether or not the saving is to be executed (save screen) is displayed on the LCD 14 as shown in Figure 7(A). Specifically, the CPU 40 reads image data of the save screen from the image data 562 stored in the ROM 56 of the game cartridge 32 according to an instruction of the user and develops the same on the VRAM 50. Then, the LCD driver 52 reads the image data of the save screen developed on the VRAM 50 according to the instruction of the CPU 40 and outputs the same on the LCD 14.

Every time the screen is displayed, such the process is executed in the following description, and therefore, a detailed description on each occasion will be omitted.

It is noted that the save screen is displayed on a part of the game screen, and strictly speaking, the image data of the save screen is overwritten to a part of the image data of the game screen developed on the VRAM 50. The same is true for a save end screen and an error screen described later.

If “NO” is selected on the save screen, it is returned to the game screen so as to continue the game. On the other hand, if “YES” is selected, a writing process of the last for writing this latest game data is executed. If succeeding in writing the last this latest game data is

successful, a screen (save end screen) indicative of a message of an end of the save (normal end) such as “SAVING ENDED” is displayed as shown in Figure 7(B). On the other hand, if failing in writing the ~~last latest~~ game data ~~fails and being the data is prohibited from being~~ written to ~~another the only other remaining~~ backup area, a screen (error screen) indicative of an alarm message such as “BACKUP MEMORY IS BROKEN DOWN. PLEASE REPAIR IT. LAST DATA IS NOT SAVED” is displayed as shown in Figure 7(C).

As described above, it is possible to securely leave the game data from at least one previous generation stored in the flash memory, and furthermore, when the alarm message is displayed, e.g. (by sending the cartridge for repair to a manufacturer), it is possible to exchange 10 with a new flash memory with the game data one generation ago left. This allows, although the game data is one generation ago, the player to resume where the player left off.

It is noted that in a case that after one backup area becomes unwritable and the other backup area is prohibited to be written, the player further advances the game, a message 15 indicative of an unwritable state is displayed at only a time that the ~~last latest~~ game data is to be stored, and therefore, there is a fear that a previous effort by the player comes to nothing. Therefore, as shown in Figure 8, a write prohibiting flag 58c is further provided within the memory 58, and a prohibiting flag of the backup area to which the overwriting is prohibited is set (turned-on), that is, overwriting in the backup area is prohibited, the alarm message can be quickly displayed with reference to the write prohibiting flag 58c at a time of the game start or 20 at a time of being instructed to display the save screen, and therefore, it is possible to eliminate the above-described wasted effort.

Furthermore, it is possible that the physical breakdown or the electrical breakdown occurs in the write prohibiting flag area 58c, and therefore, predetermined data (“FF” in this embodiment) is written to a specific cell (e.g., last cell) of each of the first backup area 58a and 25 the second backup area 58b, and when the writing to the backup area becomes disable, the predetermined data written to the specific cell of the backup area is rewritten by another data (data except for “FF”). This makes it possible to know the backup area which becomes

unwritable by searching the data written in the specific cell of each of the backup areas, and in a case one backup area becomes unwritable, it is determined that overwriting on the old game data written to the other backup area is prohibited.

It is noted that the predetermined data (“FF”) written to the specific cell is data for prohibiting the game data from being written to the cell on which writing (rewriting) of the data is not generally performed and therefore, the life thereof becomes long.

The above-described operation is processed by the CPU 40 shown in Figure 2 according to flowcharts shown in Figure 9 to Figure 12. As shown in Figure 9, when the game is started, game data generated according to the progress of the game is stored in the work memory 44 in a step S1. That is, the game data is renewed. In a following step S3, it is determined whether or not save screen display is instructed. If “NO” in the step S3, that is, the save screen display is not instructed, the process directly returns to the step S1.

On the other hand, if “YES” in the step S3, that is, the save screen display is instructed, a save screen shown in Figure 7(A) is displayed in a step S5.

In a following step S7, it is determined whether or not saving is selected. That is, it is determined whether or not “YES” is selected on the save screen. If “NO” in the step S7, that is, if “NO” is selected, the process directly returns to the step S1. On the other hand, if “YES” in the step S7, that is, if “YES” is selected, a backup process is executed in a step S9, and then, the process returns to the step S1.

It is noted that although a description is only made ~~on a case the for the situation where~~ a backup process is executed in response to an instruction ~~of saving from by~~ the user ~~to perform~~ a saving of game data as shown in Figure 9, the backup process ~~is~~ may also be automatically executed during the game. For example, a timer for counting a predetermined time is provided, and at a time that the timer counts the predetermined time, the backup process is executed, or in accordance with the progress of the game or in response to a predetermined operation (an operation except for instructing the saving) by the user during the game, the backup process is executed. These are arbitrarily determined (set) by a developer or a programmer of the game.

As shown in Figure 10, when the backup process is started, historical data is assigned to game data (~~last game data~~) stored in the work memory 44 in a step S11. That is, numerical data corresponding to a count value of the counter is assigned (included in) to the ~~last latest~~ game data. Next, the historical data included in the game data which is stored in the backup area 58a 5 and the backup area 58b are read in a step S13, and a selecting process of the backup area as the write-objective area is executed in a step S15.

In a following step S17, the ~~last latest~~ game data, i.e., latest saved game data, data and the ~~its~~ associated historical data are written to the selected write-objective backup area. That is, a writing process of the ~~last for saving this latest~~ game data including the ~~its~~ historical data is 10 executed. In a step S19, it is determined whether or not a normal writing ~~is was~~ performed by use of a checksum. That is, the CPU 40 reads the written game data from the flash memory 58 and compares the game data with the game data stored in the work memory 44 so as to determine if there is a match or coincidence. If there occurs a match, that is, if “YES” in the step S19, it is determined ~~the that a~~ normal writing ~~is was~~ performed, so a renewal process of 15 the historical data is executed in a step S21, and then, a save end screen shown in Figure 7(B) is displayed in a step S23 so as to return to the backup process.

On the other hand, in a case ~~the situation where the compared~~ game data are not coincident to each other, that is, if “NO” occurs in the step S19, it is determined ~~the that a~~ normal writing ~~is was~~ not performed, and ~~so~~ writing of the game data is ~~repeatedly~~ executed a 20 predetermined times (three times, for example) in a step S25. At this time, every time that the writing of the game data is performed, the above-described checksum is performed.

In a following step S27, it is determined whether the writing of the game data succeeded or not. If “YES” in the step S27, that is, if the writing the game data succeeded, the process proceeds to the step S21. On the other hand, if “NO” occurred in the step S27, that is, if failing 25 in ~~a normal~~ writing of the game data even if failed, even after the writing process is performed a predetermined number of times, the writing process is ended (forcedly terminated) in a step S29 so as to prohibit ~~an~~ overwriting of the game data ~~stored~~ one generation ago in a step S29.

Then, an error screen shown in Figure 7(C) is displayed in a step S31, and then, the process returns to the backup process. Thus, in ~~a case where~~ the writing process of the game data (~~last-latest~~ game data) is forcedly terminated, the ~~last~~ saving of the latest game data is abandoned or canceled.

5 It is noted that in a case of providing the write prohibiting flag area 58c in the flash memory 58 as shown in Figure 8, it is appropriate that writing of the write prohibiting flag is executed when the writing of the game data is ended in the step S29, and then, the error screen is displayed in the step S31. This makes it possible to easily know whether or not ~~the-a~~ writing of the ~~last-latest~~ game data is going to be prohibited by referring with the write prohibiting flag area at a time of starting the game or at a time that the save screen is instructed to be displayed, and therefore, it is possible to immediately display the error screen.

10 Furthermore, even if the write prohibiting flag area 58c is not provided, when the writing process of the game data is ended in the step S29, the data of the last cell of the backup area which becomes unwritable may be changed. In this case, it is possible to easily know 15 whether or not ~~the-a~~ writing of the ~~last-latest saved~~ game data is going to be prohibited by referring to the data in the last cell of each of backup areas at a time of starting the game or at a time that the save screen is instructed to be displayed.

20 As shown in Figure 11, when a selecting process of the backup area is started, each of the read historical data is compared with each other in accordance with the condition for area selection described above in the step S31. The condition for area selection utilized at this time is determined depending upon the historical data assigned to the ~~last-latest~~ game data as described by utilizing the table 1. The backup area to which the game data including the oldest (order) historical data is to be written is selected as the write-objective backup area and, the then, the selecting process is returned.

25 It is noted that in a case the game data is not stored in the flash memory 58 in an initial state, it is impossible to refer the historical data. Therefore, priorities are assigned to respective areas, and until the game data is written to all the backup areas, the game data may be written

according to the priorities. That is, it is possible to write the game data from the first backup area 58a to the second backup area 58b in this order or in reverse order thereto.

Furthermore, as shown in Figure 12, when a renewal process of the historical data is started, it is determined whether or not the historical data is “4”, that is, a count value of the counter is “11” in a step S41. If “NO” in the step S41, that is, if the count value is not “11”, the historical data within the work memory 44 is added with 1 in a step S43, that is, the counter is incremented, and then, the renewal process of the historical data is returned.

On the other hand, if “YES” in the step S41, that is, if the count value is “11”, the historical data within the work memory 44 is rewritten by “1” in a step S45, that is, the counter 10 is reset and then, the renewal process of the historical data is returned.

In at least one example implementation, when writing of the game data to one backup area becomes disabled, writing of the game data to the other back-up area is prohibited, and therefore, it becomes possible to securely leave game data from one previous generation stored in memory. That is, it is possible to prevent the previously stored game data from being 15 damaged.

In addition, the number of rewriting times of the two backup areas is made even by sequentially writing the game data to the respective backup areas, and therefore, it is possible to maximize the life of the flush memory for rewriting.

The game machine 10 of another example implementation, discussed below, is the same 20 as the above-described implementation except that three or more backup areas are formed in the flash memory 58 provided in the game cartridge 32, and therefore, a duplicated description is omitted.

As shown in Figure 13, the flash memory 58 is provided with N of the backup areas 58n (1 ≤ n (natural number) ≤ N). In this example implementation, the game data is sequentially 25 stored in respective one of the backup areas 58n in the same manner as the above-described embodiment. For example, until the game data is stored in all the backup areas 58n, the backup areas are selected from n=1 in order so as to store the game data; however, after the game data

is stored in all the backup areas 58n, the game data is overwritten according to a predetermined condition for area selection.

It is noted that the backup area is first selected from n=1 in order until the game data is stored in all the backup areas 58n in this example implementation; however, if only the writing 5 is not performed on the same backup area, the backup area may be selected by an arbitrary method.

The condition for area selection in this implementation is as shown in Table 2 (below), and the condition for area selection to be utilized is, as the above-described implementation, determined in advance in correspondence to the historical data of the lastlatest game data. For 10 example, if the historical data of the lastlatest game data is “1”, the condition for area selection utilized is $2N > 2N-1 > \dots > N+2 > N+1$.

Table 2

Historical Data of <u>Last</u> <u>Latest</u> Game Data	Condition for Area Selection
1	$2N > 2N-1 > \dots > N+2 > N+1$
2	$1 > 2N > \dots > N+3 > N+2$
3	$2 > 1 > 2N > \dots > N+4 > N+3$
•	•
•	•
N-1	$N-2 > N-3 > \dots > 1 > 2N > 2N-1$
N	$N-1 > N-2 > \dots > 2 > 1 > 2N$
N+1	$N > N-1 > \dots > 3 > 2 > 1$
•	•
•	•
2N-2	$2N-3 > 2N-4 > \dots > N-1 > N-2$
2N-1	$2N-2 > 2N-3 > \dots > N > N-1$
2N	$2N-1 > 2N-2 > \dots > N+1 > N$

That is, as the above-described embodiment, a counter capable of counting numbers 15 more than the multiples (2N) of the number of the backup areas (N) is provided, and a

numerical value of the historical data is sequentially renewed by incrementing a count value of the counter. Furthermore, when a multiple of the number of the backup areas is counted, the counter is reset and therefore, it is possible to set the condition for area selection by utilizing simple inequalities shown in the table 2.

5 In addition, similar to the above-described implementation, the game data including the historical data indicated by the numeral on the right side of the inequality sign is older than the game data including the historical data indicated by the numeral on the left side of the inequality sign. Accordingly, in a case of writing the last latest game data according to the condition for area selection, the backup area stored with the game data including the historical data of the numeral described at the right end (far-right side) is selected as the write-objective backup area.

10 It is noted that in a case the backup area which becomes unwritable exists, the historical data of the game data stored in the backup areas except for the backup area which becomes unwritable is acquired, and by applying them to the condition for area selection, the write-objective backup area is selected. Accordingly, in a case the backup area stored with the game data including the historical data having a numeral at the far-right side of the condition for area selection becomes unwritable, the backup area stored with next older game data (game data including the historical data having a numeral at a second position from the right end) is selected as the write-objective area. That is, the read historical data is compared according to 15 the condition for area selection, and consequently, the backup area stored with the game data including the historical data of the numeral at the far-right side is selected as the write-objective backup area. In other words, the backup area stored with the game data having an older time of writing to the flash memory 58 is preferentially selected.

20 Next, a description is provided of the process implemented by CPU 40; however, since the process is the same as the previously-described implementation except that a part of the backup process is changed, and so therefore, a redundant description of the process is omitted.

More specifically, the CPU 40 executes a backup process according to a flowchart

shown in Figure 14 and Figure 15. It is noted that Figure 14 and Figure 15 are applied with the same reference numerals (step numbers) as Figure 10 with respect to the same or similar process.

As shown in Figure 14, in the backup process of this embodiment, if "NO" in the step 5 S27, that is, if failing in writing of the game data, it is determined whether or not another writable backup area is present in a step S101. More specifically, it is determined whether or not two or more writable backup area exist. If "NO" in the step S101, that is, if only one writable backup area is present, the writing process of the game data is ended in the step S29 so as to prohibit the writing of the backup area.

On the other hand, if "YES" in the step S101, that is, two or more writable backup area exist, another writable (write-objective) backup area is selected in a step S103, and then, the process returns to the step S17. That is, in the step S103, a backup area stored with next older game data to the game data recorded in the backup area selected by the selecting process of the backup area in the step S15 is selected. Thus, the selecting process of the write-objective backup area is repeatedly executed until the overwriting is prohibited.
10
15

It is noted that although it is determined whether or not another writable backup area exists in the step S101 in this embodiment, the determination may also be simplified by providing the write prohibiting flag described in Figure 8.

It is noted that until the write prohibiting flag is turned-on, that is, a normal backup area 20 becomes one, the writing process of the game data to the backup area which becomes unwritable is executed, and therefore, as shown in Figure 16, by providing a writing area (unwritable flag area) 58p of the flag (unwritable flag) as to the backup area which becomes unwritable, an unnecessary writing process is avoided, and whereby, it is possible to quickly complete the backup process. In this case, when the unwritable flag area 58p is fixed, in a case 25 a physical breakdown or an electrical breakdown occurs to the area, it becomes impossible to refer to the unwritable area with ease, and therefore, it is necessary for the unwritable flag area 58p to be moved (rewritten) to another backup area 58n in a case rewriting is required in the

same manner as the backup area 58n. For example, it is appropriate that the unwritable flag area 58p is rewritten to the backup area recorded with the next older data to the game data recorded in the backup area 58 written with the last-latest game data. Accordingly, in this case, it is necessary to complete the writing process at a time that two writable backup areas are

5 present.

Furthermore, in a case where the unwritable flag area 58p is not provided, if data indicative of whether writable or unwritable is written to the end cell of each backup area, as in the above described previous example implementation, it is possible to know the unwritable backup area 58n.

10 According to this example implementation, when writing is performed to the oldest game data and writing of the game data is disabled with respect to all the backup areas but one, writing of the game data to the one backup area is prohibited, and therefore, it is possible to surely leave the game data one generation ago. That is, it is possible to prevent the game data from being damaged.

15 In addition, by sequentially writing the game data in a plurality of backup areas, the number of the writing times to the respective backup areas can be made even, and therefore, it is possible to maximize a writable life of the flush memory.

It is noted that although a description is made on the game machine of the present invention taking the portable game machine as an example in the embodiments, it is needless to
20 say that the present invention can be applied to any form of electronic device such as personal computer, game machine for home use, arcade game machine, portable information terminal, cellular phone, and etc. if only it functions as a game machine which executes a game program and saves the game data in a nonvolatile memory such as flush memory and etc.

In addition, the ROM and the nonvolatile memory stored with the backup control
25 program need not to be loaded in the game machine in a detachable manner and may be provided within the game machine.

Although the present invention has been described and illustrated in detail, it is clearly

understood that the same is by way of illustration and example only and is not to be taken by way of limitation, the spirit and scope of the present invention being limited only by the terms of the appended claims.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Patent Application of

Confirmation No. 8866

Hitoshi YAMAGAMI

Atty. Ref.: 723-1451

Appl. No. 10/712,312

TC/A.U. 3714

Filed: November 14, 2003

Examiner: Meagan J. THOMASSON

For: METHOD AND APPARATUS FOR BACKING-UP GAME DATA

* * * * *

November 24, 2008

MAIL STOP AF

Commissioner for Patents

P.O. Box 1450

Alexandria, VA 22313-1450

Sir:

SUBSTITUTE SPECIFICATION

(CLEAN VERSION)